Intelligent Energy Management:
An Algorithm to Minimize Electricity Cost

Project ID #73

Jianming Geng, Tzu-Yun Huang, Alyssa Tavares, Zhriuo Zhang

Advised by Dr. Anil Aswani

Table of Contents

Executive Summary

The integration of renewable energy into energy systems brings challenges, including low efficiency and unpredictable power output, which raise the electricity cost. Reinforcement Learning (RL) thus becomes a promising solution due to its model-free nature, combating the inherent unpredictability through the integration. The team, Energy Agent, builds upon an already-existing solution to offer a more reasonable scheduling strategy that helps minimize the electricity cost using RL. The process works as follows: two Deep Neural Networks (DNN) are first trained with Long Short Term Memory cells (LSTM) to adapt to an environment similar to the energy system. One DNN aims to learn when to use what energy resources for a household through the environment's exploration in order to save cost. The other DNN evaluates the exploration and provides feedback. The exploring agent will then discretize the value space in the environment to distinguish which actions are more cost-saving, and a Mixed-Integer Programming is employed to find the optimal actions in order to minimize scheduling cost of energy resources. The result of the exploration is noticeable: the agent with LSTM cells converges almost 2 times faster than the vanilla model without LSTM cells. And the LSTM variant is more robust against unseen data. However, the money saved is relatively unaffected by the result of the exploration and remains roughly the same across all models. It will be up to future efforts to redesign the reward logistics in order to potentially save more money with the RL models.

1. Section One: Introduction

The integration of renewable-based resources, such as solar and wind power, has significantly changed the energy system scheduling. Given that solar and wind power are unpredictable from place to place, they present a lot of challenges in determining the optimal operational schedule for the energy system. Specifically, the source of renewable energy, such as solar, fluctuates based on weather conditions, thus creating uncertainty in estimating the overall cost of the system scheduling. Traditionally, energy scheduling problems are approached by model-based methods, where mathematical models are constructed to predict energy consumption. However, as renewable-based energy becomes part of the equation, it is particularly hard for model-based methods to accurately predict energy consumption because of the unpredictable nature of those resources. Moreover, it is usually the case that there is not a clear model formulation to even start tackling the problem, hence a model-based approach could not be employed. In response to these challenges, model-free approaches become popular as time progresses. In particular, Reinforcement Learning, as an example of a model-free solution, learns optimal decisions through interaction with the environment. Unlike model-based approaches, RL does not rely on any models explicitly, which makes it relatively robust against uncertainties.

1.1. Literature Review

As mentioned, there are generally two types of approaches to the energy scheduling problem: model-based and model-free. Model-based approach emphasizes more on the definition of a precise model in order to formulate the problem in a way that the model could solve it. On the other hand, the model-free approach relies on RL (AlMahamid et al. 2021), which formulates the problem as a Markov Decision Process (MDP). However, the energy scheduling problem is slightly different from the traditional MDP in that there is an emphasis on strict enforcement of constraints (Brock et al. 2021). For example, if the energy provided is less than the energy consumed, the system may experience a shutdown due to the power imbalance. Brock also states the need for more real-time data in order to determine the real-time pricing (RTP). This reflection is important to tackle energy scheduling problems because such problems also need real-time data to determine the optimal scheduling.

1.1.1 Q-Learning and Deep Q-Learning

The technique used to determine RTP is known as Q-learning (Jang et al. 2019), which enables the agent to learn the value of an action in a particular state. Mnih further highlights the importance of Q-learning on RL agent through a demonstration of Atari game (Mnih et al., 2013). The moral of the story to take away from Atari is that an RL agent is able to deal with uncertain events successfully with Q-learning. Deep-Q learning (DQL) is then introduced based upon the development of Q-learning (Mammen et al. 2019), where the scale of the problem is no longer an issue. Q-learning is power, where every action in an infinitely sized space is well defined. For instance, an agent in Q learning has limited options to take on a chess board, and each action has a clear consequence followed by it. If the problem is scaled up infinitely such that the agent can move any piece on an infinite size chess board, then there is no way the agent will learn the best strategy in a finite amount of time.

1.1.2 Multi-Layer Perceptron and Long Short Term Memory

DQL is aware of the existence of such infinite action space, and thus proposes an estimate function to predict the possible actions and their corresponding values. Some examples of estimated functions are linear function or more complex Deep Learning function (LeCun et al. 2015). Deep Learning functions include Multi-Layer Perceptron (Popescu et al. 2009) and Long Short Term Memory (LSTM) functions (Hochreiter et al. 1997). However, even with DQL, strict enforcement of constraints could not be realized in an energy scheduling problem. This is not done until the introduction of constraint-aware RL (Yasmeena et al. 2015). The idea of renewable-energy integration into the traditional energy grid further changes the landscape of the field (Qiu et al. 2023). Qiu investigates the electric vehicle and its particular contribution in Power Systems. Hou combines all sorts of ideas (Hou et al. 2023), such as DQL and renewable-energy integration, into a single framework called Mixed Integer Programming-Deep-Q Learning (MIP-DQN). The framework has two phases: the first phase is to train a RL agent, and the second phase is to enforce the constraint through mathematical programming. The framework is optimized in order to minimize the scheduling cost, yet the model could still be improved structurally.

1.2. Contributions

In order to improve the MIP-DQN framework, as mentioned in the Literature Review section, this paper makes several changes in order to improve the performance of the model (potentially to save more money). The first change is during the construction of the training environment (details of the term will be explained in the later section). Instead of sampling actions, a Large Language Model (LLM) is employed to simulate the process and thus increase the randomness in the process. Another change of the framework is to replace Multi-Layer Perceptron (MLP) with Long Short Term Memory (LSTM) cells. The performance of the modified framework is then compared with the vanilla MIP-DQN, MIP-DQN with variant exploration noise, and MIP-DQN with more nodes in MLP function. To sum up, the main contributions of the paper are (1) simulation of data within the training environment using LLM and (2) an incorporation of LSTM cells into the estimate function to potentially make the agent memory-aware for better performance.

2. Section Two: Methodology and Outcome

2.1. Brief Intro of Mixed Integer Programming-Deep-Q Network

2.1.1 Dataset

The dataset was utilized from *Optimal Energy System Scheduling Using A Constraint-Aware Reinforcement Learning Algorithm* Paper Research. The dataset included the energy consumptions in the construction in both traditional and renewable energy, which detailed across various time intervals and pricing. Time series is one of the most critical aspects of this dataset, which is essential for capturing the inherent variability and stochastic nature of renewable energy sources, such as solar and wind. In addition, the dataset is important in the training and evaluation of the Mixed Integer Programming-Deep-Q Network (MIP-DQN) algorithm. By observing the energy consumption patterns and capturing the variability and uncertainty that characterizes the renewable energy sources, the dataset can be simulated and analyzed. The MIP-DQN model can be trained and validated under closely related real-world scenarios of the energy system because of the depth of the dataset, which can further facilitate the development of an adaptable model.

2.1.2 Algorithm Framework

The MIP-DQN algorithm is designed to minimize the energy scheduling cost of each household. For instance, if the cost of using electricity at 5:00PM in the afternoon is $10 using traditional power source, while the cost is $5 in the morning using traditional power source, then it is intuitive to use traditional power source in the morning and solar panel in the afternoon. In addition to the optimal strategy that uses lower cost sources depending on times of the day, the algorithm also trades electricity surplus whenever the demand is smaller than the supply. ***Figure 1*** shows the logistics and purpose of the algorithm.

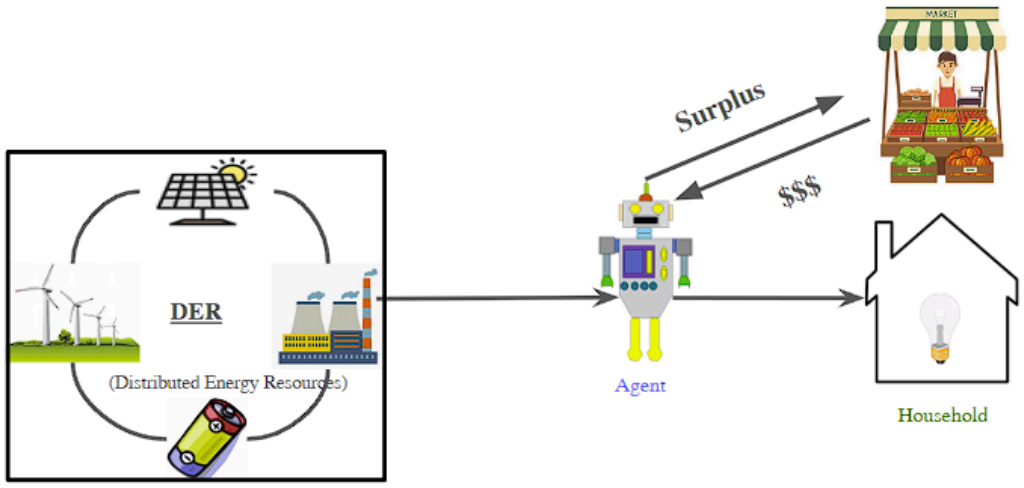


Figure 1. *The working logistic of the MIP-DQN algorithm.*

The MIP-DQN algorithm has two phases: training and deployment. The end goal of the training phase is to estimate the parameter θ of the estimate function Q(θ). During the deployment phase, Q(θ) is used to make decisions for the optimal scheduling while sticking to the constraint (such as flow balance of energy).

The training process involves two sub-processes: construction of an environment and training the estimate function. The definition of environment is left in the later section to introduce, while the training of an estimate function is straightforward: it involves training two models concurrently: one model corresponds to the estimate function, which explores the environment and finds the available actions for each time step. The other model evaluates the exploration and provides feedback to the first model. Then, the first model updates the parameter θ iteratively until the loss function is converged. Convergence of the loss function means that the function is able to assign values to each action. For example, if the action "use traditional power source to generate electricity at 10:00AM" corresponds to a cost of $10 while "use solar panel generated electricity at 10:00AM" corresponds to a cost of $5, then the estimate function will assign a higher value to the solar panel action, meaning that this is a better action to take. The algorithm is as follows:

$$min_{\theta} \sum_{i=1}^{|B|} (r_{t,i} + \boldsymbol{\gamma}\, Q_{\theta}^{\,target}\ (s_{t+1,i},\ argmax_{a} Q_{\theta}(s_{t+1,i},\ a))\ -\ Q_{\theta}(s_{t+1,i},\ a_{t,i}))^{2},$$

where $r_{t,i}$ is the reward at time stamp t of the i^{th} simulated trial within the environment, $s_{t+1,i}$ is the state, and $a_{t,i}$ is the action.

The deployment process involves an employment of a Mixed Integer Programming solver, which will specify the objective and the constraint. Under the previous assumption, the objective is to minimize the cost of energy, while the constraint is to ensure flow balance of energy.

2.2. Improvements on the algorithm

2.2.1. Simulated Environment

A simulated environment is added to make the training model more robust. By definition, in the energy scheduling context, an environment just means an isolated place that only consists of traditional power plants, solar panels, the demand of electricity, and the corresponding price of electricity. During the training phase mentioned above, there are two RL models, where one explores the environment and one evaluates the exploration. Now, imagine the two models as two babies, and one baby gets into the environment without knowing anything about the energy scheduling. As the baby explores, it gradually learns the price of electricity at different times of the day, and it also learns when to use what sources (traditional or solar) in order to minimize the cost. As the baby finishes exploring, the other baby starts to evaluate how the strategy of the exploring baby does in terms of saving money. Then, the second baby would give feedback to the first baby, and the first baby would re-explore the environment. The logistic flow of how the environment interacts with two models are given in ***Figure 2***.

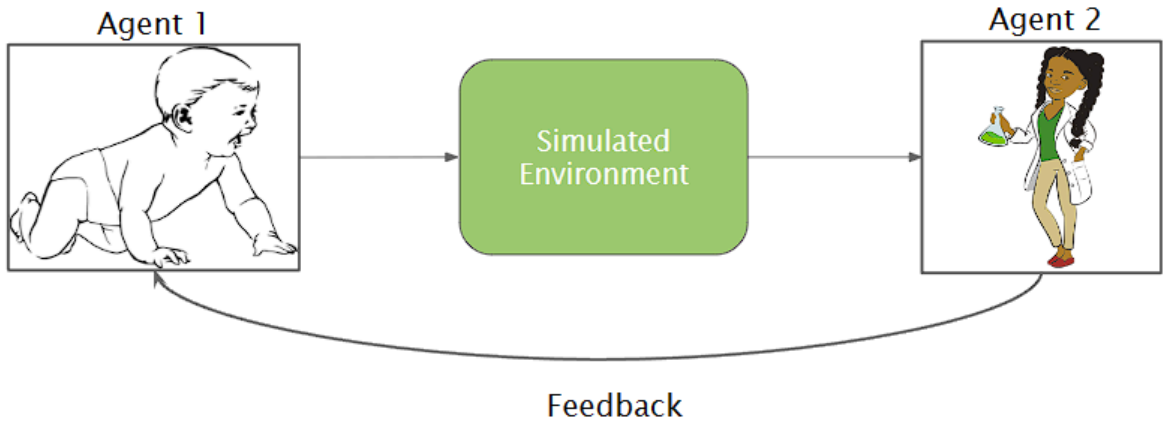


Figure 2. *logistic flow of how the environment interacts with two agents*

A process called simulated environment is thus introduced, which is a reference to a power grid system in the real world. A power system works in a dynamic equilibrium between supply and demand – the electricity generation must match with the demand all the time, not more or less by too much. Given the fact that conventional energy sources are adjustable, and that the demand, or more professionally, the load, is not controllable but predictable, any power system must be able to forecast the load, such that the system operators could adjust and schedule the power output to meet the demand at any time. An Automatic Generation Control (AGC) is the subsystem responsible for this – it gathers and analyzes energy information, makes demand forecasts, and then automatically adjusts the power output, in response to changes in the load. In order to operate a machine learning model, it is very important to properly define a simulated

environment, in that it is where the defined agent receives information and acts on it to make the agent itself more robust.

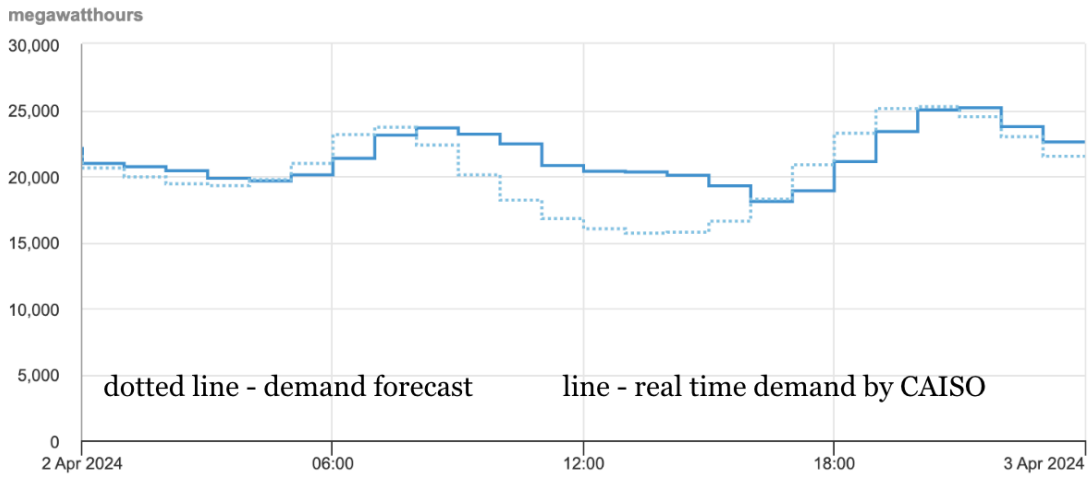
2.2.2 Long Short Term Memory Cell

During the training phase, there are two models, where one explores the environment to learn the optimal strategy, while the other evaluates the exploration. In the original construction, both models are Multi-Layer Perceptron (MLP). In essence, MLP has several layers, and each layer is just a linear function plus an activation function. There are some benefits with this structure, including its simplicity and fast training time, among others. However, there are some key drawbacks to a MLP, namely that it is not complex enough to reflect the nature of many real-life problems. In consideration of the energy scheduling problem, it is the case. Yet the data itself is a time-series type, which indicates that the current state depends on one or more past events. Thus, it is natural to try to tackle the problem with Long Short Term Memory (LSTM) cells. The logic behind it is that LSTM could put more emphasis on the more recent events so that the agent can make decisions with its recent past experiences. Although there is no theoretical guarantee that a LSTM cell is needed in an estimate function, it otherwise shows that an inclusion of LSTM cells could boost the performance of the model in its exploration.

2.3. Results

2.3.1. Simulated Environment

This section evaluates the performance of this simulated environment, myAGC, including accuracy of predictions, reliability, and sustainability.



The graph shows a comparison between the actual demand recorded by CAISO on 4/2, 2024 and the prediction made by myAGC. The average error is approximately 12%, and given that the industrial standard is 10-15%, this section concludes that the simulated environment is reliable for generating simulated data during the construction of the environment, thus making the training process more smooth.

2.3.2. Exploration Loss

The exploration loss reflects how well the model explores the environment. If the loss is high, it means that the model does not fully understand the environment. In other words, the model might still use sources of electricity with higher cost when it is obviously not the optimal action to take. By contrast, if the loss is low, it means the model does a good job exploring. ***Figure 3*** shows how the model with LSTM cells performs in exploration compared to other variants based on the same framework.

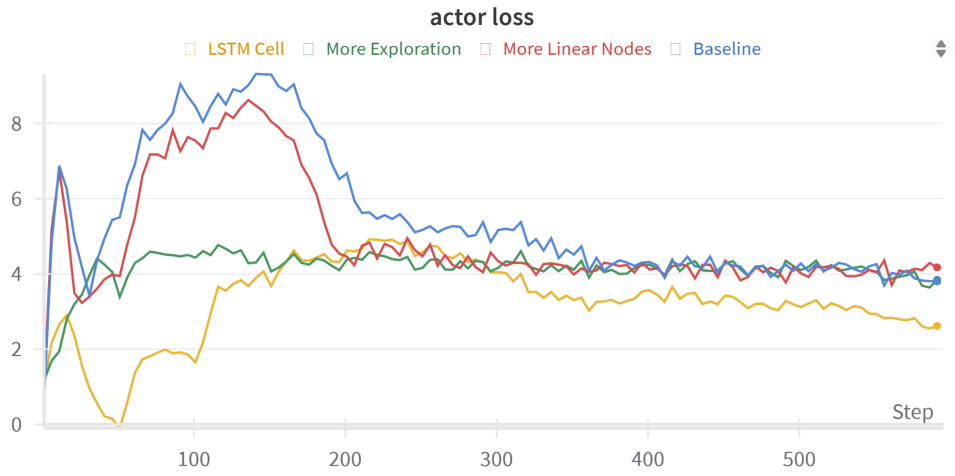


Figure 3. *The loss for different variants of MIP-DQN during exploration. The lower the better.*

Figure 3 shows the loss across four different variants of MIP-DQN: (1) vanilla MIP-DQN, (2) MIP-DQN with more nodes in its estimate function, (3) MIP-DQN with higher tendency to explore instead of taking the greedy action, and (4) MIP-DQN with LSTM cell in its estimate function. At the beginning, LSTM and More Exploration variants start with lower loss. When the iteration step reaches 150, there is a huge jump for the other two variants. This jump indicates that they are exploring an environment they have not experienced before. By comparison, LSTM and More Exploration variants do not react to the unseen environment, which shows that they are robust against unseen data. Towards the end, the LSTM variant reaches a loss closer to 2, which is almost half the loss of other variants. This result demonstrates two advantages of the LSTM variants: (1) MIP-DQN with LSTM cell is robust to unseen environment, which makes it a good model to combat volatility; (2) MIP-DQN with LSTM cell converges almost 2 times faster than all the other variants, making it a very strong learner.

2.3.3. Operation Cost/ Reward

The operation cost is measured by the mean episode reward. A relatively high operation cost corresponds to a smaller reward (more negative), which means that the model does not adapt the optimal strategy as there is still another strategy that leads to lower cost. By comparison, a relatively low operation cost corresponds to a larger reward (closer to 0), meaning that the model is able to perform a close-to-optimal strategy to save money. Figure 4 shows how the model with LSTM cells performs in reward compared to other variants based on the same framework.

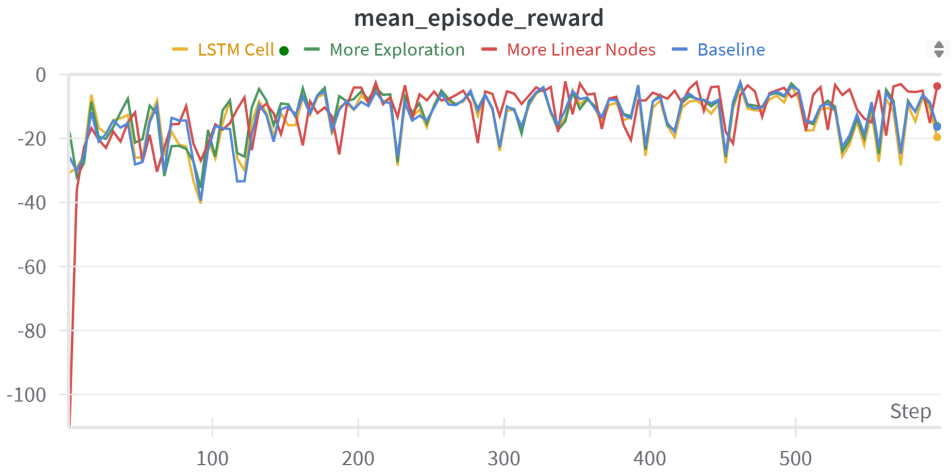


Figure 4. *The reward for different variants of MIP-DQN during operating. The lower the better.*

Figure 4 shows the reward across four different variants of MIP-DQN during operating: (1) vanilla MIP-DQN, (2) MIP-DQN with more nodes in its estimate function, (3) MIP-DQN with higher tendency to explore instead of taking the greedy action, and (4) MIP-DQN with LSTM cell in its estimate function. It could be seen that the performance of all variants is very similar, a stark contrast to their performance in exploration. This indicates that no matter how well a model learns about an environment, there is always a bottleneck in how much money the model can save given some strategy. This result alludes that there is possibility to further minimize the operation cost if a redesign of the reward function takes place.

2.4. Future Steps

2.4.1. Redesign of Reward Function

In order to have a lower operation cost, the model needs to perform with a reward closer to 0 from the negative end. However, given the current setting of the reward function, no matter how

well the model learns about the environment and optimal strategy, it will never be able to make a breakthrough on the reward. This indicates that the reward function is not closely related to the exploration, which should never be the case. One way to make sure a reward function is good or bad is to perform a statistical analysis between the reward and the exploration. Once the correlation between the two factors are positive, this is an indication that the new reward function might be usable in the training of a better model.

2.4.2. Other Applications

In addition to the energy scheduling problem, there are other applications where the current framework could be applied or adapted easily. Electric Vehicles (EV) is another application that was explored with a DQN reinforcement learning model to determine how system demand can be integrated with the scheduling of electric vehicle charging. To implement this, the hours left to charge can be estimated given specifications provided for the battery's usable capacity and the voltage of a level 2 charger ("2023 Tesla Model 3 Performance AWD - Specifications."). In addition, data on system demand from January 2021 to July 2021 is provided by CAISO OASIS. A more simple DQN model was constructed to estimate the best time for a person to begin charging their EV based on the beginning amount of charge left in their vehicle. A low amount of demand at a given hour could serve as a price signal for EV charging. A result from this DQN model indicates that between 4 a.m. to 5 a.m. would be the best time to begin charging. This is another topic for further exploration by integrating renewable resources (such as in our MIP-DQN model) that could serve as an additional price signal in the case for finding the optimal time for electric vehicle charging.

3. Conclusion

This paper has proposed several changes/improvements to the original MIP-DQN framework, including simulation of training data, which increase the robustness of the model against unseen data, and integration of LSTM cells, which increase the efficiency of the exploration, in order to minimize electricity cost. The performance of the revised model is compared with three other variants: (1) vanilla MIP-DQN, (2) MIP-DQN with more nodes in its estimate function, and (3) MIP-DQN with higher tendency to explore instead of taking the greedy action. It turns out that LSTM based MIP-DQN performs the best in its exploration of the environment, and it is robust to unseen data. However, the four models are roughly equivalent in terms of their operational cost. Some of the future steps include a revision of the reward function and to apply the revised framework to some other problems. This field is an exciting topic that can allow one to evaluate how to minimize utility cost and also use renewable sources and apply this to a machine learning technique.

References

AlMahamid, F. and K. Grolinger, "Reinforcement Learning Algorithms: An Overview and Classification," 2021 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), ON, Canada, 2021, pp. 1-7, doi: 10.1109/CCECE53047.2021.9569056.

Hochreiter, S. and J. Schmidhuber, "Long Short-Term Memory," in Neural Computation, vol. 9, no. 8, pp. 1735-1780, 15 Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

Hou, S., P. P. Vergara, E. Duque, and P. Palensky. Optimal energy system scheduling using a constraint-aware reinforcement learning algorithm. 05 2023.

Jang, B., M. Kim, G. Harerimana and J. W. Kim, "Q-Learning Algorithms: A Comprehensive Classification and Applications," in IEEE Access, vol. 7, pp. 133653-133667, 2019, doi: 10.1109/ACCESS.2019.2941229.

LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. Nature 521, 436–444 (2015). https://doi.org/10.1038/nature14539

Mammen, P. Mary and H. Kumar. Explainable ai: Deep reinforcement learning agents for residential demand side cost savings in smart grids. 10 2019.

Mnih, V., K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. 12 2013.

Popescu, Marius-Constantin & Balas, Valentina & Perescu-Popescu, Liliana & Mastorakis, Nikos. (2009). Multilayer perceptron and neural networks. WSEAS Transactions on Circuits and Systems. 8.

Qiu, Dawei, Wang, Yi, Hua, Weiqi, and Strbac, Goran. 2023. “Reinforcement Learning for Electric Vehicle Applications in Power Systems: A Critical Review,” Renewable and Sustainable Energy Reviews, vol. 173, 113052. ISSN 1364-0321. doi: https://doi.org/10.1016/j.rser.2022.113052.

“2023 Tesla Model 3 Performance AWD - Specifications.” *EV Specifications*, 11 Apr. 2024, evspecifications.com/en/model/186229d.

Yasmeena, S., and G. Tulasiram Das. 2015. “A Review of Technical Issues for Grid Connected Renewable Energy Sources.” International Journal of Energy and Power Engineering 4

(5): 22. https://doi.org/10.11648/j.ijepe.s.2015040501.14.